



National Aeronautics and Space Administration

Engineering Elegant Systems: Systems as Mathematical Categories

1 March 2019

Michael D. Watson, Ph.D.

Consortium Team

UAH

George Washington University

Iowa State

Texas A&M

Dependable System Technologies, LLC

Multidisciplinary Software Systems

Research Corporation (MSSRC)

Missouri University of S&T

University of Michigan

AFRL Wright Patterson



Space Launch System





Theoretical Basis of Systems Engineering

Understanding Systems Engineering



- ◆ **Definition – System Engineering is the engineering discipline which integrates the system functions, system environment, and the engineering disciplines necessary to produce and/or operate an elegant system.**

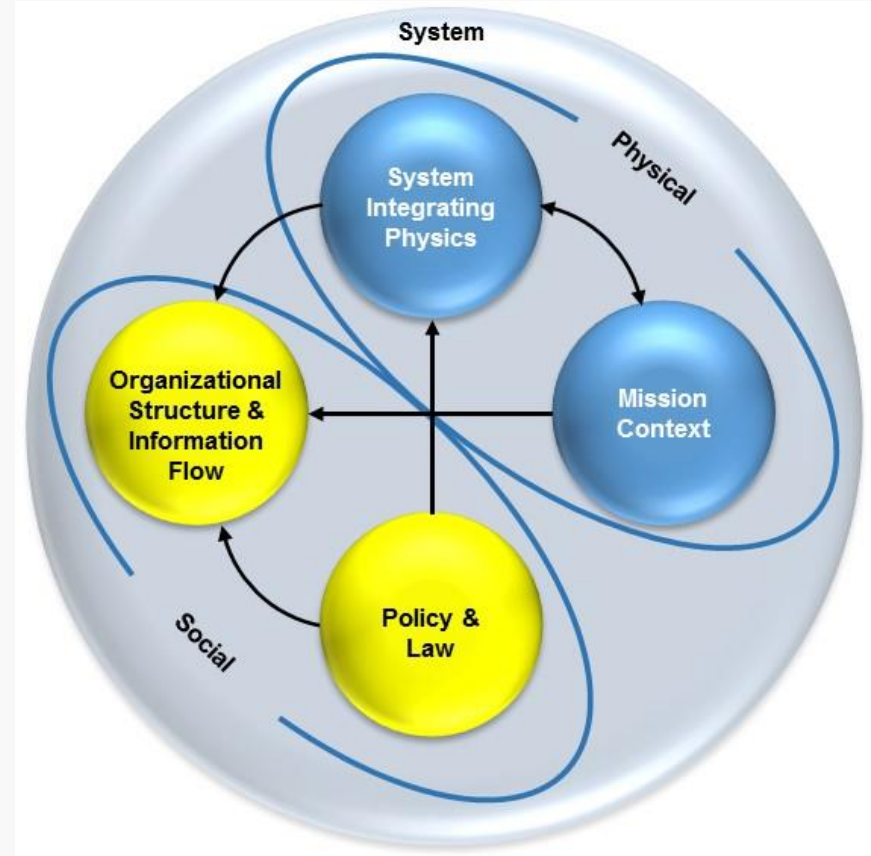
- **Elegant System** - A system that is robust in application, fully meeting specified and adumbrated intent, is well structured, and is graceful in operation.

- ◆ **Primary Focus**

- **System Design and Integration**
 - Identify system couplings and interactions
 - Identify system uncertainties and sensitivities
 - Identify emergent properties
 - Manage the effectiveness of the system
- **Engineering Discipline Integration**
 - Manage flow of information for system development and/or operations
 - Maintain system activities within budget and schedule

- ◆ **Supporting Activities**

- **Process application and execution**
 - Processes organize the engineering



Systems Engineering Postulates



System Integration (physical/logical system)

Discipline Integration (social system)

Both System and Discipline Integration

- ◆ **Postulate 1: Systems engineering is system specific and context dependent in application**
- ◆ **Postulate 2: The Systems Engineering domain consists of subsystems, their interactions among themselves, and their interactions with the system environment**
- ◆ **Postulate 3: The function of Systems Engineering is to integrate engineering disciplines in an elegant manner**
- ◆ **Postulate 4: Systems engineering influences and is influenced by organizational structure and culture**
- ◆ **Postulate 5: Systems engineering influences and is influenced by budget, schedule, policy, and law**
- ◆ **Postulate 6: Systems engineering spans the entire system life-cycle**
- ◆ **Postulate 7: Understanding of the system evolves as the system development or operation progresses**
- ◆ **Postulate 7 Corollary: Understanding of the system degrades during operations if system understanding is not maintained.**

Systems Engineering Principles



- ◆ **Principle 1: Systems engineering integrates the system and the disciplines considering the budget and schedule constraints**
- ◆ **Principle 2: Complex Systems build Complex Systems**
- ◆ **Principle 3: A focus of systems engineering during the development phase is a progressively deeper understanding of the interactions, sensitivities, and behaviors of the system, stakeholder needs, and its operational environment**
 - Sub-Principle 3(a): Mission context is defined based on understanding of the stakeholder needs and constraints
 - Sub-Principle 3(b): Requirements and models reflect the understanding of the system
 - Sub-Principle 3(c): Requirements are specific, agreed to preferences by the developing organization
 - Sub-Principle 3(d): Requirements and design are progressively elaborated as the development progresses
 - Sub-Principle 3(e): Hierarchical structures are not sufficient to fully model system interactions and couplings
 - Sub-Principle 3(f): A Product Breakdown Structure (PBS) provides a structure to integrate cost and schedule with system functions
 - Sub-Principle 3(g): As the system progresses through development, a deeper understanding of the organizational relationships needed to develop the system are gained.
 - Sub-Principle 3(h): Systems engineering achieves an understanding of the system's value to the system stakeholders
 - Sub-Principle 3(i): Systems engineering seeks a best balance of functions and interactions within the system budget, schedule, technical, and other expectations and constraints.

Systems Engineering Principles



◆ Principle 4: Systems engineering has a critical role through the entire system life-cycle

- Sub-Principle 4(a): Systems engineering obtains an understanding of the system
- Sub-Principle 4(b): Systems engineering defines the mission context (system application)
- Sub-Principle 4(c): Systems engineering models the system
- Sub-Principle 4(d): Systems engineering designs and analyzes the system
- Sub-Principle 4(e): Systems engineering tests the system
- Sub-Principle 4(f): Systems engineering has an essential role in the assembly and manufacturing of the system
- Sub-Principle 4(g): Systems engineering has an essential role during operations, maintenance, and decommissioning

◆ Principle 5: Systems engineering is based on a middle range set of theories

- Sub-Principle 5(a): Systems engineering has a physical/logical basis specific to the system
- Sub-Principle 5(b): Systems engineering has a mathematical basis
- Sub-Principle 5(c): Systems engineering has a sociological basis specific to the organization(s)

◆ Principle 6: Systems engineering maps and manages the discipline interactions within the organization

◆ Principle 7: Decision quality depends on system knowledge present in the decision-making process

◆ Principle 8: Both Policy and Law must be properly understood to not overly constrain or under constrain the system implementation

Systems Engineering Principles



- ◆ **Principle 9: Systems engineering decisions are made under uncertainty accounting for risk**
- ◆ **Principle 10: Verification is a demonstrated understanding of all the system functions and interactions in the operational environment**
- ◆ **Principle 11: Validation is a demonstrated understanding of the system's value to the system stakeholders**
- ◆ **Principle 12: Systems engineering solutions are constrained based on the decision timeframe for the system need**
- ◆ **Principle 13: Stakeholder expectations change with advancement in technology and understanding of system application.**
- ◆ **Principle 14: The real physical system is the perfect model of the system**
 - Kullback-Liebler Information shows the actual system is the ideal information representation of the system
 - $I(f, g) = \int f(x) \log(f(x)) dx - \int f(x) \log(g(x|\theta)) dx = 0$

System Engineering Hypotheses



- ◆ **Hypothesis 1: If a solution exists for a specific context, then there exists at least one ideal Systems Engineering solution for that specific context**

- Hamilton's Principle shows this for a physical system

$$-\int_{t_1}^{t_2} (\delta T - \delta V + \delta W) dt = 0$$

- ◆ **Hypothesis 2: System complexity is greater than or equal to the ideal system complexity necessary to fulfill all system outputs**

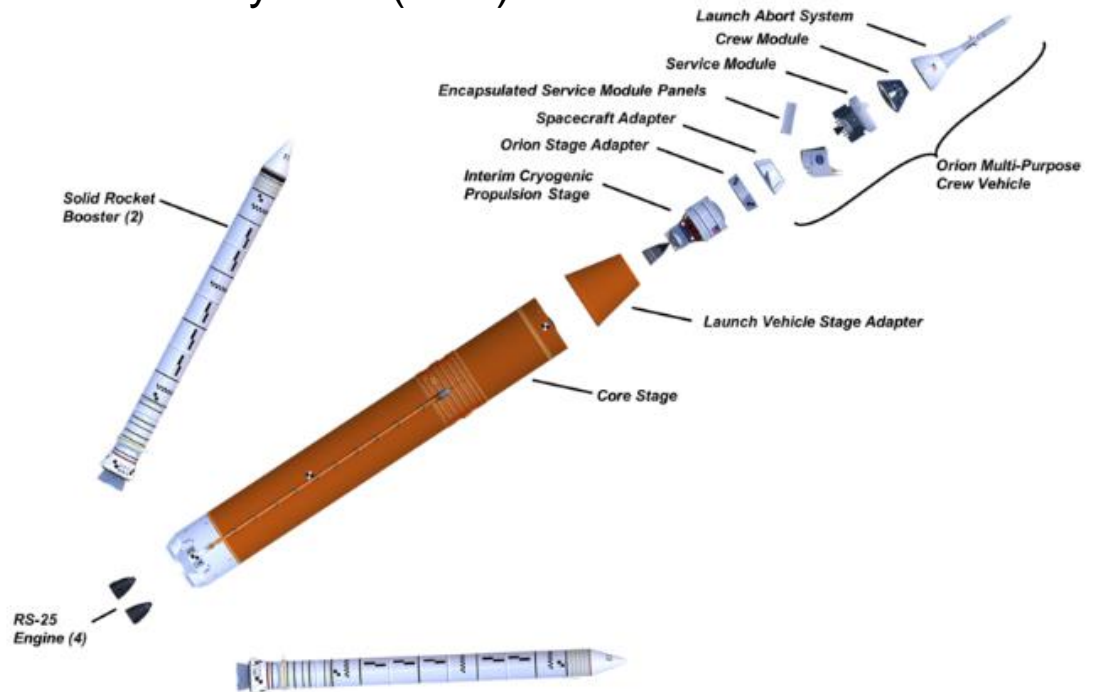
- ◆ **Hypothesis 3: Key Stakeholders preferences can be accurately represented mathematically**

Mathematical Basis of Systems Engineering: Mathematical Category Theory

◆ Systems are comprised of 2 basic structures

- Postulate 2: The Systems Engineering domain consists of subsystems, their interactions among themselves, and their interactions with the system environment
- Components
- Relationships among components
 - Physical
 - Logical
- Relationships with the environment
 - Physical

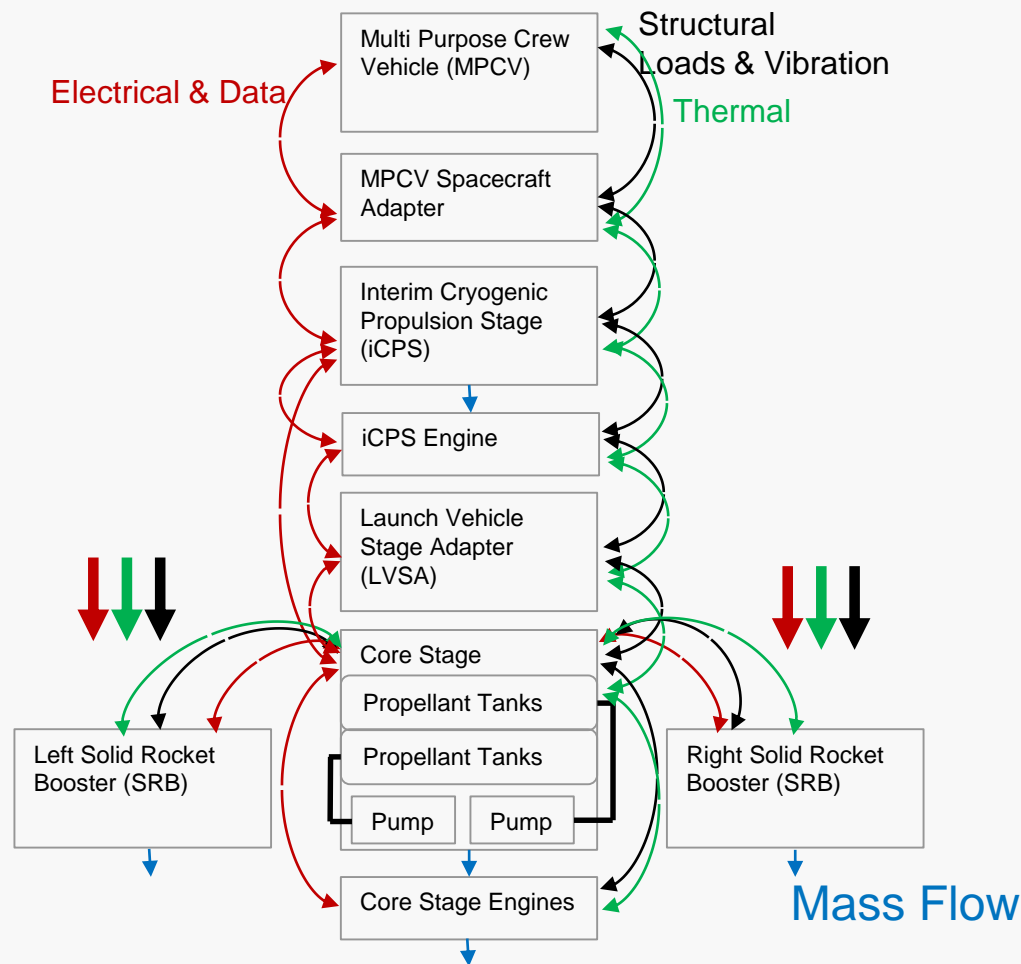
Major Components of the NASA Space Launch System (SLS)



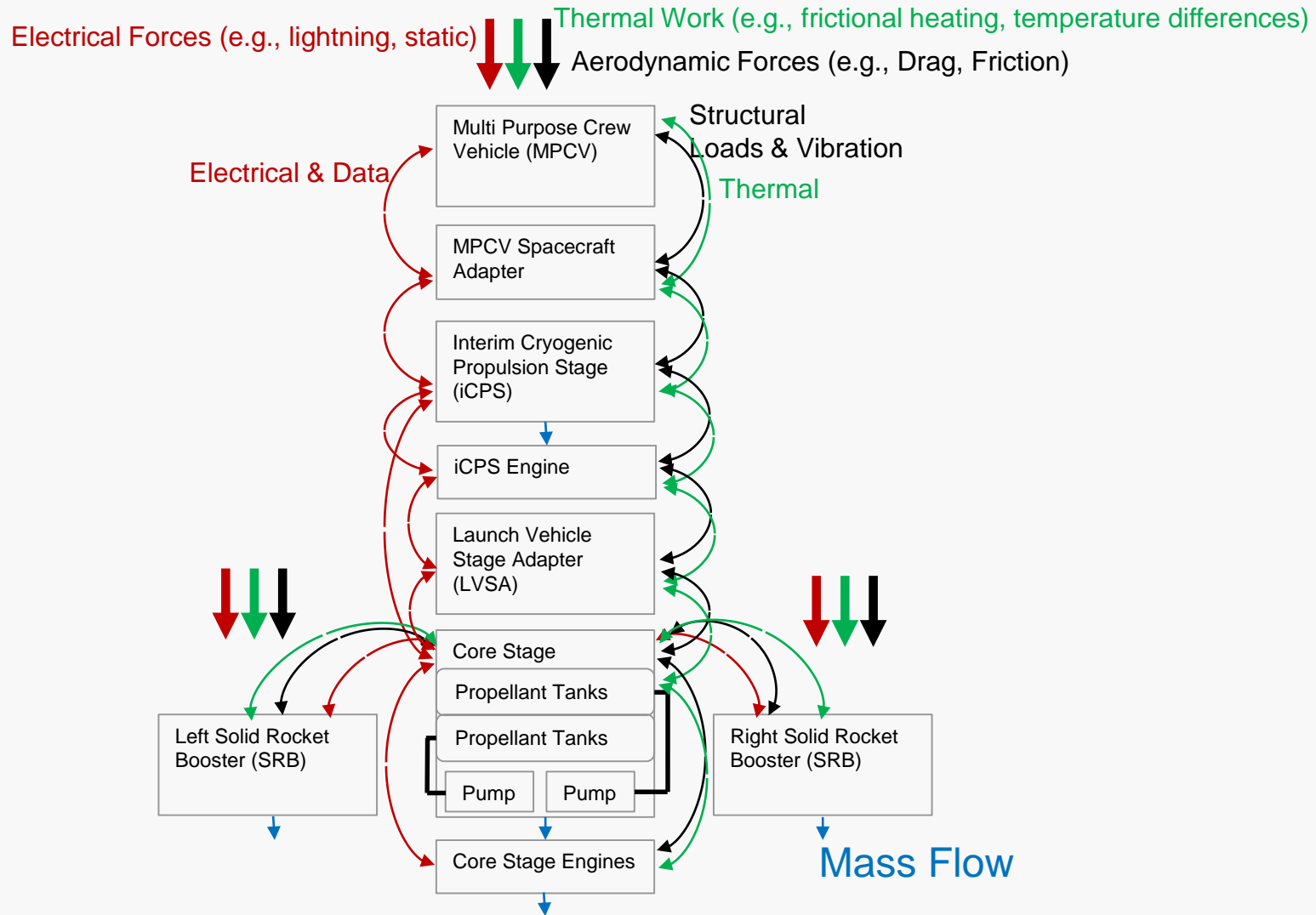
Rocket Physical and Logical Relationships



Electrical Forces (e.g., lightning, static) Thermal Work (e.g., frictional heating, temperature differences)
Aerodynamic Forces (e.g., Drag, Friction)



Rocket as a Mathematical Category



◆ A Mathematical Category consists of

- Objects (i.e., system components): a, b, c, \dots
- Arrows (i.e., system relationships between components and the environment): f, g, \dots

◆ A Mathematical Category has properties

- Domain/Codomain

- $f: a \rightarrow b$ where a is the domain of f and b is the codomain of f

- Identify Relationship

- $\text{id}_a = 1_a: a \rightarrow a$



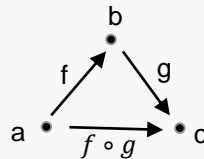
- Associativity

- $f \circ (g \circ h) = (f \circ g) \circ h$

- Composition

- Composition can be performed by various mathematical operations (i.e., addition, subtraction, multiplication, division)

- $a \xrightarrow{f} b \xrightarrow{g} c = a \xrightarrow{f \circ g} c$



Mathematical Category Types



◆ Category Types

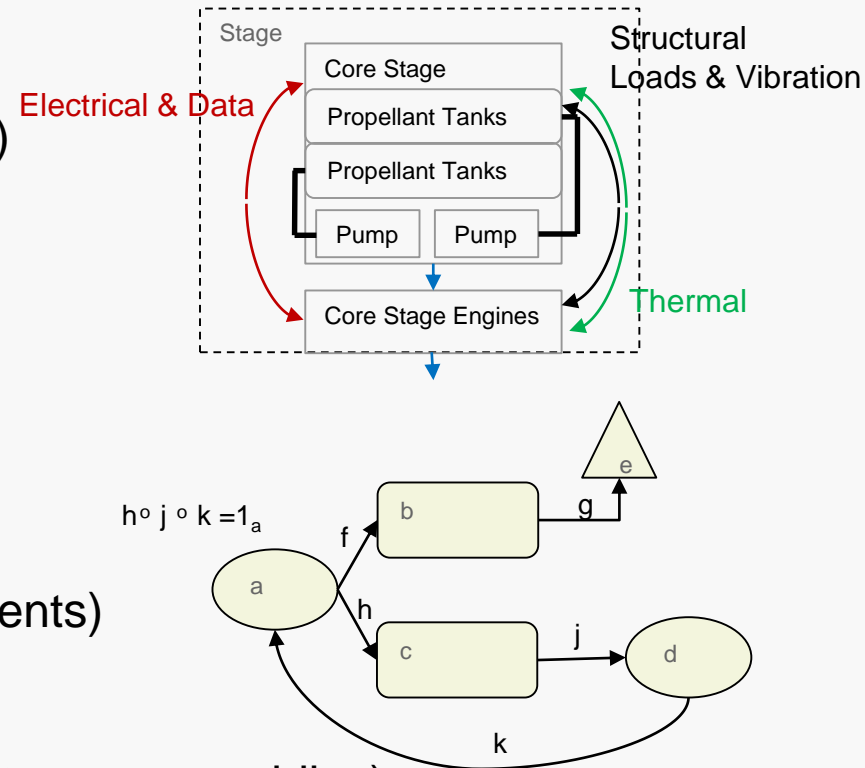
- Category of Sets
- Category of Arrows (objects are implied)
- Category of Groups
- Category of Categories
- Universal Category
- Category of Small Categories
- Abelian Categories

◆ Objects within a category can be

- Objects (i.e., individual parts or components)
- Sets (i.e., sets of individual parts)
- Groups
- Smaller Categories (i.e., stages, subsystems, assemblies)

◆ Directed Graphs

- Directed graphs, when they meet the property conditions, are a form a mathematical category



◆ Functors

- Mathematical morphisms between categories, $F: A \xrightarrow{F} C$
- Creates a mapping from one category to another
- Includes composition in the mapping

◆ Natural Transformations

- Transformation is the same among all objects
- Is commutative
- If invertible, then is a 'natural equivalence' or 'isomorphism'

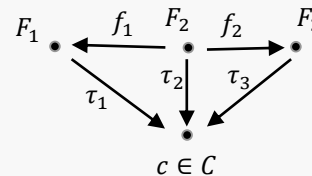
◆ Isomorphism

- If the relationships (arrows) are invertible between two objects, then the objects are isomorphic, $a \cong b$
 - $a \xrightarrow{f} b \xrightarrow{g} a$, $f = g'$, $g = f'$
- Categories can be isomorphic, $A \cong B$
 - The objects can be different, but the relationships between the objects of the two categories are preserved
 - i.e., different copies of the same system are isomorphic
 - Or, two different designs of the same system type may be isomorphic (e.g., different automobile makes with similar models)

◆ Co-cones/Co-limits

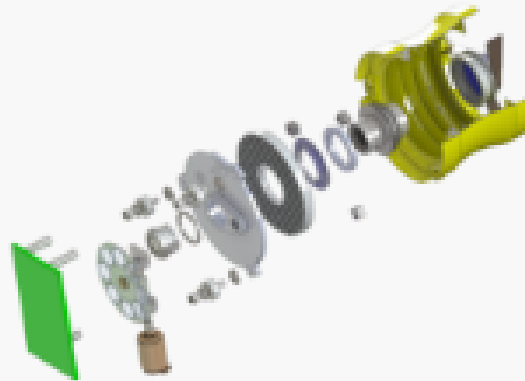
- Co-cone

- A common codomain for Functors operating on Category C



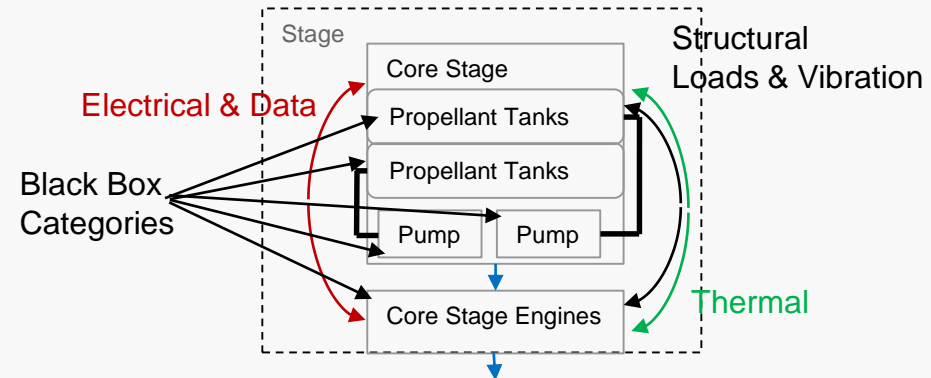
- Co-limit

- The limit of the Co-cone defining the conditions where all Functors and mappings to objects of the Category, C , are included



◆ Black Box

- Since a Category may contain smaller Categories, then an engineering 'black box' is a Category treated as an object within a larger Category



◆ System Completeness

- The mathematical structure of the system Category provides a mechanism to construct a completeness proof for a given system

◆ System Specification

- The System objects and relationships form the basis of the system requirements
- The Category must contain the correct and complete objects and relationships
 - Variations result in a system different than intended

◆ System Assembly

- Co-cones and co-limits define the assembly operations needed to construct the system Category
- The Functors map parts from the parts category(s) to the system category
 - The parts may map to sub-categories (i.e., assemblies and subsystems) within the system category
- The limits define what must be included at each step of the assembly in order to be complete

- ◆ **System Engineering Postulates provides a mathematical definition of a system**
 - Postulate 2: The Systems Engineering domain consists of subsystems, their interactions among themselves, and their interactions with the system environment
- ◆ **System are Mathematical Categories**
 - Mathematical Category Theory provides the mathematical structure to fully represent a system: all of its components and all of its physical and logical relationships
- ◆ **Mathematical Category Theory provides insight into systems**
 - Category Types
 - Categories as objects within larger Categories
 - Directed Graph Representations
 - Functors
 - Natural Transformations
 - Isomorphism
 - Co-cones and co-limits
- ◆ **Mathematical Category Theory supports representations of systems**
 - Engineering Black Box
 - Parts/components, assemblies, subsystems as smaller categories within the system category
 - System Completeness
 - System Specification
 - System Assembly